

**COMPREHENSIVE GIS BASED IMPLEMENTATION
STRATEGY FOR ENABLING DYNAMIC RIDE SHARING
AND CAB HAILING**



FINAL YEAR PROJECT UG 2012

By

NUST201200751BIGIS10412F - Syed Ramiz Sami

NUST201200275BIGIS10412F - Sarmad Ali Qureshi

NUST201200629BIGIS10412F - Shahid Nawaz Khan

NUST201201096BIGIS10412F - Zunerah Sanai

**Institute of Geographical Information System
School of Civil and Environmental Engineering
National University of Sciences and Technology, Islamabad, Pakistan**

CERTIFICATE

Certified that the contents and form of Final Year Project entitled “**Comprehensive GIS based implementation strategy for enabling dynamic ride sharing and cab hailing**” submitted by Syed Ramiz Sami, Sarmad Ali Qureshi, Zunerah Sanai, and Shahid Nawaz Khan have been found satisfactory for the requirement of the degree.

Supervisor: _____

Lecturer (Ms. Quratulain Shafi, IGIS-NUST)

External Examiner: _____

Name: _____

Designation: _____

DEDICATED TO OUR PARENTS AND TEACHERS

ABSTRACT

Travelling in the urban areas could at times be a herculean task due to the overburden on public transport and high rates of taxi. There is a dire need of an efficient system that will be able to accommodate people along with the convenience and reliability. “A comprehensive GIS based implementation strategy for dynamic ride sharing and cab hailing”, is a service which will address those issues of transport which were previously left unaddressed. It is a cross platform compatible system, having the ability to work on different devices with ease. The main aim of the project is to provide a reliable cab sharing system which will be able to reduce total distance covered by the vehicle along with cost by allowing customers with same route and time window to share cabs. The data used for the project was acquired from OSM (Open street maps) an open source volunteered Geographic information system from which users can upload and download the data depending on their needs. The database of the system resides on a remote server which accepts requests through a user interactive user interface and then makes decision based on different algorithms and user needs. The system has two main interfaces that is the web and android interface. Different algorithms have been tested on the database depending on the type of request some of which are, Dijkstra Shortest path algorithm, A* for optimal routing and TSP (travelling salesman problem) for finding out route for more than two points. The algorithms in the start phase gave satisfactory results and with the passage of time through artificial intelligence the system will be able to learn and make the system more efficient on the usage of data accumulated on daily basis and running artificial algorithms on it. The system will be able to cater the user needs in an urban environment, in a country where such concept is naïve.

ACKNOWLEDGEMENTS

We would like to show our gratitude to Almighty Allah for giving us knowledge, power and strength to accomplish this task. We learnt a lot while doing this project and this will certainly help us in our forth coming life.

We would like to thank our supervisor Ms.Quratulain Shafi, who was very supportive throughout the project. Her supervision helped us a lot during the times of difficulties.

We wish to thank our teachers for their detailed suggestions and insights of their thoughtful teaching style on the overall development of **Comprehensive GIS Based Implementation Strategy for Enabling Dynamic Ride Sharing and Cab Hailing.**

Syed Ramiz Sami

Sarmad Ali

Shahid Nawaz

Zunerah Sanai

LIST OF TABLES

Table 1: PHP vs Python vs Ruby	20
Table 2: Custom vs CMS vs MVC	22
Table 3: Django vs Pyramid vs Flask	23
Table 4: Spatial Lookups in Spatial Databases	24
Table 5: Spatial Functions possible in Spatial Databases	25

LIST OF FIGURES

Figure 1: Locating objects using GPS	11
Figure 2: Workflow.....	19
Figure 3: Tools and Technologies.....	26
Figure 4: System Architecture	29
Figure 5: Data Preparation	34
Figure 6: Creating Routing System	39
Figure 7: Self Learning of road networks.....	40
Figure 8: Location and speed calculated every 5 seconds	41
Figure 9: Location and speed sent to server.....	42
Figure 10: Timestamp assigned to information received at server	43
Figure 11: Road selected from Location Information.....	44
Figure 12: Harmonic mean taken.....	45
Figure 13: Values updated using harmonic mean.....	46
Figure 14: Ride Matching Algorithm	47
Figure 15: ER Diagram.....	50
Figure 16: Sign in page.....	51
Figure 17: Signup Page.....	51
Figure 18: Edit profile page	52
Figure 19: Admin home page	52
Figure 20: Roads editing page	53
Figure 21: Page to digitize roads	53
Figure 22: A request, as viewed in the admin panel	54
Figure 23: Application Homepage.....	55
Figure 24: Setting time on the application for time window	56
Figure 25: Picking origin and destination points on the application	57
Figure 26: Scheduling rides on the application.....	58

TABLE OF CONTENTS

ABSTRACT.....	4
ACKNOWLEDGEMENTS.....	5
LIST OF TABLES.....	6
LIST OF FIGURES.....	7
TABLE OF CONTENTS.....	8
CHAPTER 1.....	10
INTRODUCTION.....	10
1.1 Background.....	10
1.2 Literature Review.....	18
CHAPTER 2.....	19
MATERIALS AND METHODS.....	19
2.1 Market Research.....	20
2.2 Tools and Technologies.....	26
2.3 Server Side Development.....	29
2.4 Preliminary Data Preparation.....	32
2.5 Algorithm Development.....	40
2.6 Data Model Development.....	49
CHAPTER 3.....	51
RESULTS.....	51
CHAPTER 4.....	59
RECOMMENDATIONS.....	59
REFERENCES.....	60

LIST OF ABBREVIATIONS

1	OGC	Open Geospatial Consortium
2	GIS	Geographical Information System
3	GPS	Global Positioning System
4	JSON	Javascript Object Notation
5	API	Application Program Interface
6	MVC	Model View Controller
7	DSF	Django Software Foundation
8	CMS	Content Management System
9	XML	Extensible Markup Language
10	HTML	Hypertext Markup Language
11	WWW	World Wide Web
12	CSS	Cascading Style Sheets
13	DBMS	Database Management System
14	OSM	Open Street Map
15	TSP	Traveling Sales Person
16	JWT	JSON Web Token
17	HTTP	Hyper Text Transfer Protocol
18	JS	Java Script
19	VGI	Volunteered Graphic Information

INTRODUCTION

1.1 Background

The word ‘real-time’ refers to the actual time in which a process is taking place. With the advancement in telecommunication and modern electronics field, most of the data transmission not only has become online but is also providing the users with real-time access to useful information, along with adding more product proficiency and resourcefulness to the analyses which are associated with it. Different analysis related to navigation, weather forecasting, natural hazard and disaster management are now being done in real-time environment enabling the analysts to study and analyze the phenomena in a dynamic way where it can bring efficiency and enhance the confidence level in the results from the analysis. With the advancements in GIS and remote sensing, the urge to perform geospatial analysis increased, resulting in the incorporation of such tools and techniques in almost every field. GPS is one of the most useful inventions of the 20th century which not only changed the perception of getting accuracy in location acquisition but also contributed a lot in the different domains of science and technology. Global Positioning System is a satellite based navigation system which was designed for military and intelligence purposes at the climax of Cold War in 60s. GPS consists of a network of satellites, orbiting the Earth in their destined paths, beaming down signals on the Earth which can be detected through a GPS receiver. These signals carry the time (determined through high precision clocks on-board) at which the signal transmit the satellite and the geographical data, enabling the users to get their exact location, velocity as well as elevation anywhere and anytime on the planet. In 1960, US navy tested the GPS satellites launched by the USA. There were 5 satellites, orbiting the Earth, which helped the ships in navigating the oceans to correct their location according to the planned path after every hour. In 1967, previous system was succeeded by the Timation satellites. Timation satellites were launched and developed by the National Research Laboratory. The main

objective behind Timation was the transmission of precise time reference using highly accurate clocks to send ranging signals on the ground. With the success of timation, GPS developed rapidly. Between 1978 and 1985, 11 satellites were launched and GPS was widely used for military purposes. In the summer of 1993, US launched their 24th Satellite into the orbit which completed the modern constellation of GPS satellites. Today, 32 active satellites are in the orbit and are being widely used for a variety of purposes.

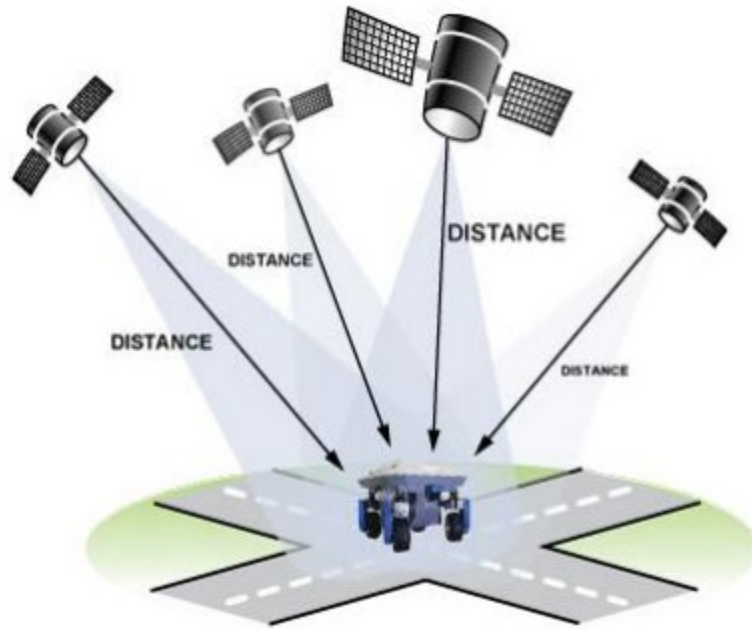


Figure 1: Locating objects using GPS

Figure 1 illustrates how satellites work to locate an object. The precision of the location depends on the satellites which are relaying signals to the receiver. Increase in the number of satellites results into accurate locations. Another factor that can affect the accuracy of location determination is the receiver type. Typical cellular phones with GPS have accuracy ranging from 8 meters to 600 meters depending on the cellular position or wi-fi positioning which would be covered later in this report. Differential GPS or DGPS, on the other hand is an enhanced form of Global Positioning System that provides location with improved accuracy in the range of 15 meter to 10 cm depending on its best utilization methods.

Our aim is to develop a cross platform compatible system which will be able to provide the ease of dynamic ride sharing, carpooling, and cab-hailing. Keeping in mind the current

needs and difficulties. This project is being designed to take care of one of the major needs of today. It will help reduce fuel cost and consumption, save time, increase the safety level, which will consequently lead to lesser air pollution and a greener environment.

To provide a better understanding of the work, below are brief descriptions of some important terms used in the project:

1.1.1 Carpooling

Carpooling (also known as auto sharing, lift-sharing, ride-sharing and covoiturage), is the sharing of auto adventures so that more than one individual can travel in a single auto.

By having more individuals utilizing one vehicle, carpooling decreases every individual's travel expenses, for example, fuel expenses, tolls, and the anxiety of driving. Carpooling is likewise an all the more ecologically agreeable and practical approach to go as sharing excursions lessens carbon discharges, movement clog on the streets, and the requirement for parking spots.

1.1.2 Continuous ridesharing

Continuous ridesharing (otherwise called moment ridesharing, element ridesharing, specially appointed ridesharing, on-interest ridesharing, and element carpooling) is an administration that masterminds one-time shared rides without prior warning. This kind of carpooling by and large makes utilization of two late innovative advances:

- GPS route gadgets to decide a driver's course and orchestrate the common ride
- Smartphones for a voyager to ask for a ride from wherever they happen to be

These components are facilitated through a system administration, which can promptly handle the driver installments and match rides utilizing an advancement calculation.

Similar to carpooling, continuous ridesharing is elevated as an approach to better use the vacant seats in most vehicles, bringing down fuel utilization and transport costs as a result. It can serve territories not secured by an open travel framework and go about as a travel

feeder administration. Ridesharing is likewise equipped for serving one-time trips, not just intermittent or planned drive excursions.

1.1.3 E-hailing or taxicab hailing

E-hailing or taxicab hailing is a procedure of requesting an auto, taxi, limousine, or any other type of transportation using virtual gadgets: PC or cell phone.

In figuring, cross-stage, multi-stage, or stage independent, is presented to PC programming or registering techniques and ideas that are executed and between work on different PC stages. Cross-stage programming may be partitioned into two sorts; one requires individual building or assemblage for every stage that it underpins, and the other one can be straightforwardly keep running on any stage without unique planning, Cross-stage projects may keep running on the same number as of every single existing stage, or on as few as two stage

For the completion of the tasks, the following technologies were used:

1.1.4 Android OS

Android OS was developed by Google and it is based on the Linux kernel. This OS was mainly designed for touchscreen mobile devices and tablets. Different functions are written, focused on the specific tasks, to get the desired results out of it. Classes and functions are written depending on the requirement to accomplish the tasks. Different APIs (Application Program Interface) which are the set of protocols and routines for application development are also incorporated in the java file of the project. The other branch of android is Extensible Mark-up Language (XML) is used for designing the interface of the application which includes icons, buttons, views and toolbars. XML is also used for the styling part which includes colors, background, themes and other styling techniques. Apart from these, GoogleMap API was used for the basemap in the application.

1.1.5 HTML

HTML or Hypertext Markup Language, is a standardized system for tagging text files to achieve font, graphic, color, and hyperlink effects on World Wide Web (WWW) pages. It

is the most popular scripting language for creating web pages and showing them. It is a language with predefined tags. Each HTML tag defines different parts of a document like head, body, title etc. There are different tags which provide different functionalities such as acronym tag etc. All the browsers are capable of reading and interpreting HTML. In short HTML is the language of browser. The frontend webpage for the project was created in HTML 5.

1.1.6 CSS

CSS stands for Cascading Style Sheets. CSS is a web scripting language that is used to define the layout of HTML pages. Features that CSS covers include fonts, margins, images, height and width and advanced styling techniques. CSS is now supported by all the browsers. There is a difference between HTML and CSS as both are confused by the users in their working. HTML is used to structure contents in specific tags, each with its own specific functionality. However, CSS focuses mainly on the styling of the web pages.

1.1.7 Javascript

Javascript is the most popular programming language of the world. It defines how a webpage behaves. HTML and CSS are used to define how a webpage looks like Javascript helps to get the answer to the question that what does a webpage do. JavaScript is a programming language used to make web pages interactive. It runs on the client device and doesn't require constant downloads from the website. The script is read by the client from the website and those scripts are then executed by the browser.

1.1.8 Web hosting server

It is a kind of Internet hosting service which allows organizations and individuals to provide accessibility to their website using the World Wide Web. Companies that facilitate with the space on a server owned or leased for use by clients are known as Web hosts. They also provide Internet connectivity, usually in a data center.

1.1.9 Python

It is a commonly used high-level, interpreted, general-purpose, dynamic programming language. Its design idea focuses on code readability. Comparatively, Python provides a better syntax that allows programmers to convey concepts in lesser lines of code than in languages like Java or C++. This language offers constructs intended to allow understandable programs on both small and large scale.

Python supports several programming paradigms, together with object-oriented, functional and imperative programming or procedural styles. It has an outsized and comprehensible standard library, along a dynamic type system and automatic memory management.

1.1.10 Django

It is an open-source web framework, written in Python, that follows the model–view–controller (MVC) architectural pattern. It is managed by a self-regulating organization recognized as a non-profit, known as Django Software Foundation (DSF).

Django's main purpose is to ease the formation of complex, database-driven websites. Django put emphasis on reusability and "pluggability" of elements, rapid development, and the rule of "don't repeat yourself". It also provides an interface for optional administrative create, read, update and delete functions, that is generated dynamically through introspection and constructed via admin models. The language Python works throughout Django, even for files, settings, and data models.

1.1.11 MVC (Model–view–controller)

MVC is a software architectural design usually used for implementing user interfaces on computers. It splits a given software application into three interconnected parts, so as to separate internal representations of information from the ways that information is presented to or accepted from the user.

1.1.12 PostGIS

PostGIS is an open source software program that adds support for geographic objects to the PostgreSQL object-relational database. PostGIS follows the Simple Features for SQL specification from the Open Geospatial Consortium (OGC).

1.1.13 Web server

This is a computer that provides web pages. Each and every web server has an IP address and perhaps a domain name. For example, entering the URL <http://www.webopedia.com/index.html> in your browser, sends a request to the web server which has a domain name webopedia.com.

1.1.14 Cloud server

It is a logical server that through a cloud computing platform, is assembled, hosted and delivered over the Internet. Cloud servers have and show comparable capabilities and functionality to a usual server but are accessed remotely from a cloud service provider. The other name for a cloud server can be virtual server or virtual private server.

1.1.15 RESTful web services

REST stands for Representational State Transfer. It is an architectural style for networked hypermedia applications, and is mainly used to build web services that are manageable, and accessible. A service based on REST is known as RESTful service.

1.1.16 Database

A database management system (DBMS) is a computer software application that cooperates with the user, other applications, and with database to collect and analyze data. A normal or usual DBMS is designed to permit the definition, creation, querying, update, and administration of databases.

1.1.17 Spatial Database

A spatial database, or geodatabase is a database that is developed to store and query data that symbolizes objects defined in a geometric space. Most spatial databases allow to represent simple geometric objects like points, lines and polygons.

1.1.18 Web development frameworks

A software framework called web framework (WF) or web application framework (WAF), is designed to support the development of web applications including web services, web resources and web APIs. Web framework's objective is to improve the overhead associated with common activities performed in web development.

1.1.19 Content Management System

A content management system (CMS) is a computer application that helps to create and modify digital content using a mutual user interface and thus usually supporting multiple users working in a cooperative environment.

1.1.20 Custom development

Custom software (also called bespoke software or tailor-made software) is a software that is particularly developed for some specific organization or another user.

1.1.21 JQUERY

jQuery is a library of Javascript that is fast, small, and feature-rich. It makes things such as HTML file traversal and management, event handling, animation, and Ajax a lot simpler with an easy-to-use API that works across a host of browsers.

1.1.22 Bootstrap

Bootstrap is a free and open-source front-end web framework for designing websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Unlike many web frameworks, it concerns itself with front-end development only.

1.1.23 Leaflet

Leaflet is a widely used open source JavaScript library used to build web mapping applications. First released in 2011, it supports most mobile and desktop platforms, supporting HTML5 and CSS3. Along with OpenLayers, and the Google Maps

API, it is one of the most popular JavaScript mapping libraries and is used by major web sites such as FourSquare, Pinterest and Flickr.

Leaflet allows developers without a GIS background to very easily display tiled web maps hosted on a public server, with optional tiled overlays. It can load feature data from GeoJSON files, style it and create interactive layers, such as markers with popups when clicked.

1.2 Literature Review

The research conducted in this paper formally defines dynamic or real-time ride-sharing, identifies optimization problems for finding best sets of ride-share matches in a number of operational scenarios, develops approaches for solving ride-share optimization problems, and tests the concepts via a simulation study of work trips in the Atlanta metropolitan area. (Wang, X., 2013)

In this paper, an integrated system for the organization of a car pooling service is shown, using several recent Information and Communication Technologies (ICT's) technologies: web, GIS and SMS. The main part of the system is an optimization module that solves heuristically the specific routing problem. (Calvo, R. W., de Luigi, F., Haastrup, P., & Maniezzo, V., 2004)

This paper examines the value of real-time traffic information to optimal vehicle routing in a nonstationary stochastic network. (Kim, S., Lewis, M. E., & White III, C. C., 2005)

MATERIALS AND METHODS

The development of this project took place in a number of important and essential steps. To give a comprehensive illustration, the following flowchart in Figure 2, has been constructed:

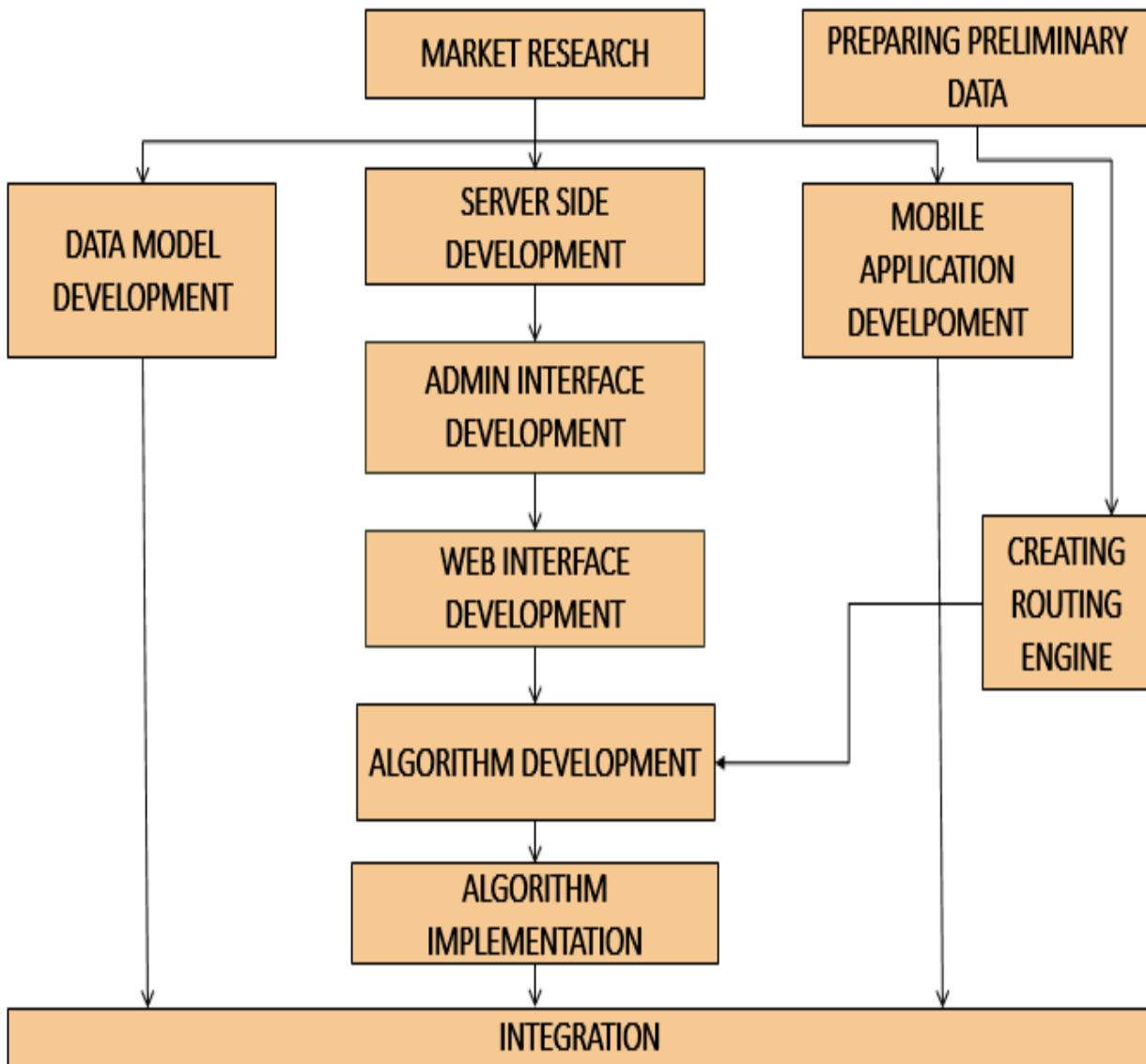


Figure 2: Workflow

2.1 Market Research

To start a project, and to make it go smoothly, it is necessary to begin with understanding its needs and requirements, and the environment it can be developed in, and the things needed to be aggregated to form the most productive output. To get all these details, Market Research was carried out to choose the best technologies for our project. Out of the variety of technologies available, few selections were made, which seemed the most useful for this application.

Language

Language is the building block of any application or program. Choosing one is not an easy task. Firstly, it is important to know that what are the pros and cons of any language, and then we select the most suitable for any specific purpose. Language requires time to be learned, so it extends the duration of the progress if the learning time is long. Python fits in the criteria of most suitable language in our application. It's the easiest to learn and implement, and it is flexible as well. It has the ability to fasten up the processing more than any other web development language, and has gained a lot of popularity in research community due to this. In our application, GIS has a major role, which can be easily incorporated using Python. It supports a vast number of libraries that can help add up GIS related functionalities.

Table 1 shows the comparison:

Table 1: PHP vs Python vs Ruby

	PHP	Python	Ruby
Purpose	Web Development	General purpose with emphasis on productivity and code readability	Making programming fun and flexible for the programmer

Readability and usability	Follows a classic approach	Most readable programming language.	Elegant, powerful, and expressive
Learning	Easy	Easiest	Hardest
Popular sites	Facebook, Wikipedia, Udemy	Google, Youtube, Disqus, Instagram, Pinterest, Dropbox	Twitter, Github, Soundcloud
Run Time (average)	4.1	1.8	2.9
GIS support	Lowest	Highest	Intermediate

Development model

Development model provides us with rules to develop software. However, not all models give us the flexibility to create the system as required. MVC (model view controller) helps us to develop application by making the best use of our knowledge. It gives us a complete control without any restrictions on creativity. It provides us with the ability to create any application according to our business logic and according to the way we like; we don't need to find plug-ins of exact functionalities. It has good built-in flexible APIs and offers an ease to change the code. Additionally, it has an easy maintenance. All these abilities make MVC a convenient and foolproof development model. On the other hand, CMS can be used only for general purposes, as it is made for all users and is resistant to changes.

Good programming practices cannot be implemented with CMS, and it does not allow customizability according to the requirements of the business logic. Rarely, but another issue that occurs in CMS is that if a performance problem arises in the core, it is very difficult to be fixed. The differences made MVC an easy and obvious choice for the development model of our project.

Table 2 shows the comparison:

Table 2: Custom vs CMS vs MVC

	Custom	CMS	MVC
Development Speed	Slowest	Fastest	Moderately Fast
Learning speed	Slow	Fastest	Slowest
Learning requirements	Programming language	Framework	Language + Framework APIs
Customizability	Very High	Lowest	Very High
Coding required	Too much	Almost none	Moderate
Popular usage	PHP	Wordpress, Joomla, Drupal	Laravel, Django, Rails
Security	Depends on programmer	High	High, but still depends on programmer
Scope	Supports everything	Supports typical applications	Supports everything

Framework

A framework can be explained as a layered structure that can help to understand which programs can be built together to produce a useful web application, and how they will interrelate. Such framework must be able to link a number of components; a connection to database, a template rendering system, a means to map urls to views, any kind of authentication system, and other components. A framework is usually more comprehensive than a protocol and more prescriptive than a structure. We have a few options when it comes to frameworks, but in order to select the best one, here are the differences. Talking about Pyramid first, it has a lot of flaws. At the very least, the online support available for Pyramid is very low and the range of libraries it supports is quite less in number. Next is Flask. Flask cannot scale and bear computational burden, as it is designed for minimal applications. Its support also doesn't fit the help requirement of a programmer beginning to code in a MVC system. At last we have Django which we decided to incorporate in this project. Django offers amazing features for admin interface. This feature allows us to generate an administration site directly from an application's model. It shortens the time required to code everything right away by putting the snippets together which you write down in Django. It doesn't take the complete control, and instead asks for your preference of models to give accessibility via the administration interface. It also has a GIS framework called geodjango incorporated in it by default which supports a lot of GIS applications. These things make Django the super-feasible choice for the project.

Table 3 shows the comparison:

Table 3: Django vs Pyramid vs Flask

	DJANGO	Pyramid	Flask
Popularity	Highest	Lowest	Lowest

Online support	110,000 StackOverflow questions	1700 StackOverflow questions	10,000 StackOverflow questions
Inbuilt functionalities	Very much	Little	moderate
Maturity	High	High	Low
GIS Support	Geographic framework inside	Moderate	Moderate

Database

As a Spatial Database, PostGIS was used for the complete package it provides. Following are the operations it can perform in comparison to other spatial databases.

- Spatial Lookups

These are the spatial lookups possible with different databases, shown in Table 4.

Table 4: Spatial Lookups in Spatial Databases

LOOKUP TYPE	PostGIS	Oracle	MySQL	Spatialite
Bbcontains	✓		✓	✓
Bboverlaps	✓		✓	✓
Contains	✓	✓	✓	✓

Coverdby	✓	✓		
Crosses	✓			✓
distance_gt	✓	✓		✓
Left	✓			
Right	✓			
Within	✓	✓	✓	✓
overlaps_above	✓			
overlaps_below	✓			
strictly_above	✓			

- Database Functions

Table 5 shows the database functions that are possible with different spatial databases

Table 5: Spatial Functions possible in Spatial Databases

FUNCTION	PostGIS	Oracle	MySQL	Spatialite
Area	✓	✓	✓	✓
AsGeoJSON	✓			✓
AsGML	✓	✓		✓

AsKML	✓			✓
BoundingCircle	✓			
Centroid	✓	✓	✓	✓
ForceRHR	✓			
Scale	✓			✓
GeoHash	✓			
Intersection	✓	✓		✓
MemSize	✓			
Translate	✓			✓

2.2 Tools and Technologies



Figure 3: Tools and Technologies

2.2.1 AJAX:

To update the web page without refreshing it again and again

2.2.2 JQuery:

It was used on the web front-end to create UI extensions

2.2.3 Mailgun:

It was used for sending emails.

2.2.4 PYTHON:

It was the development language used for the project.

2.2.5 GOOGLE MAPS:

They were used for representation on the android side.

2.2.6 LEAFLET:

Leaflet is an open source mapping library; it was used for showing maps on the web front end.

2.2.7 PostgreSQL:

It is an open source relational database; our data was stored in PostgreSQL.

2.2.8 JAVA:

It was used for the development of android application.

2.2.9 HTML:

It was the scripting language used for creating web frontends.

2.2.10 JAVASCRIPT:

It was used as the programming language at the web client side.

2.2.11 CSS:

It was used for styling the web frontend.

2.2.12 BOOTSTRAP:

It is an extension of CSS that was used for styling web frontend.

2.2.13 PostGIS:

It is an OGC compliant spatial database and the spatial extension of PostgreSQL, our spatial data was stored in PostGIS.

2.2.14 UBUNTU:

It is the operating system that was used for the project. The system was deployed on Ubuntu server 14.04.

2.2.15 MICROSOFT AZURE:

It is the cloud service by Microsoft. A D1_V2 cloud VM instance was used to deploy the project.

2.2.16 PILLOW:

It is the python imaging library. It was used for manipulating user's profile pictures.

2.2.17 ANDROID STUDIO:

It was used as the development IDE for developing Android application.

2.2.18 GOOGLE API:

Google APIs were used for getting results for routing and used as a standard.

2.2.19 DJANGO:

Django was the MVC framework used for our project.

2.2.20 DJANGO REST Framework:

This was a sub framework of Django. It was used for deploying REST based web services.

2.2.21 PGROUTING:

Pgrouting is the routing extension of PostGIS. It was used to make the routing engine.

2.2.22 JSON:

It is a light weight data format. It was used as the data transfer format on the REST.

2.3 Server Side Development

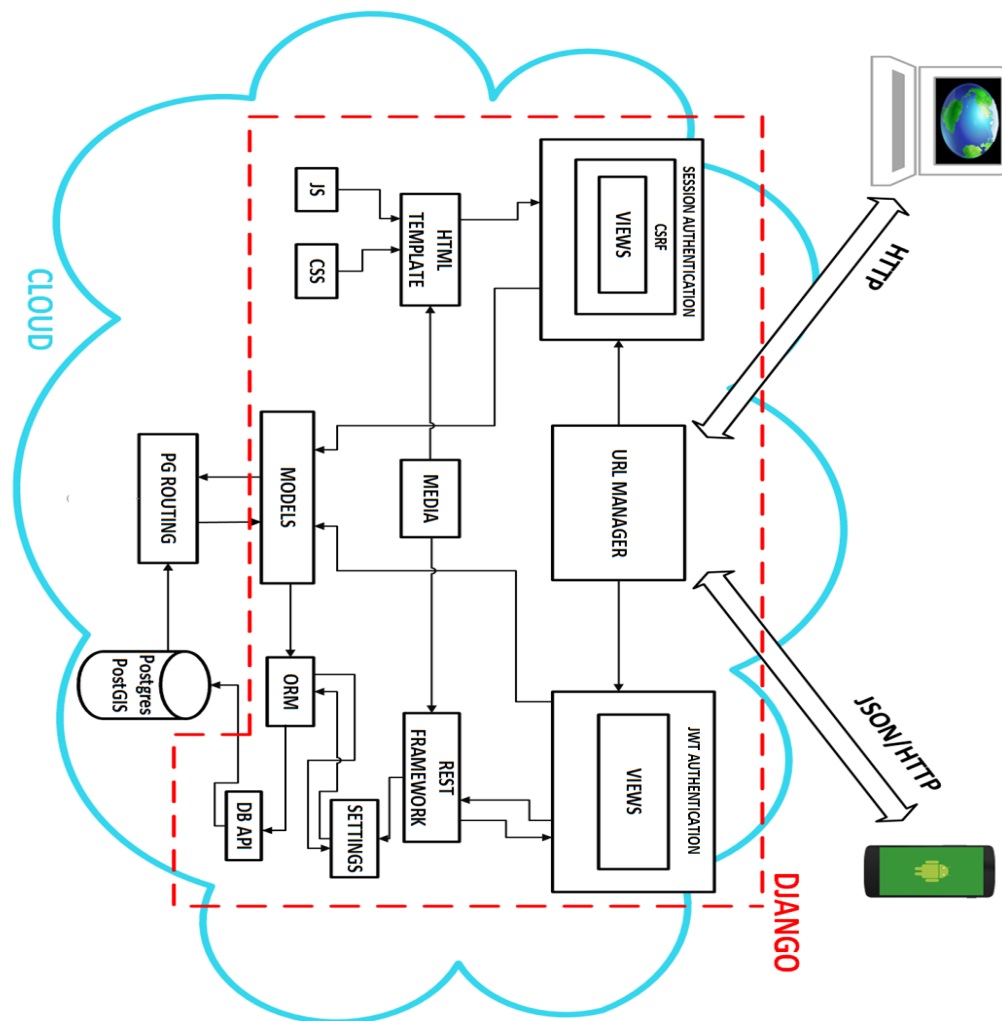


Figure 4: System Architecture

The system in Figure 4 is based on thin client architecture where most of the work is done on the server side. The server is based on Django 1.9.5 which is an MVC environment and all the modules are not dependent on the design of each other and work solely with the input that is given to them and produce outputs as required. The system is deployed on a Microsoft Azure cloud VM of D1_V2 instance which has 3.5 GB ram, 1 core of D_V2 series processor and 50 GB solid state drive (SSD). The system is running on Ubuntu Server 14.04. For the database, PostgreSQL, an open source database, is used along with its spatial extension PostGIS that is used for storing geographic data. Another extension of PostgreSQL and PostGIS i.e. Pgrouting reference is used for managing the road network of Islamabad and solve routing problems. The website and the Android application are acting as the frontends of the system and logics are handled at the server side.

The frontends are linked to the URL manager. The purpose of the URL manager is to establish a connection between the server and the frontends. URL manager is a kind of diverter. Each request received at the URL Manager is then forwarded to appropriate views. There are two kinds of views. One view is interacting with the web interface and the other with the android application through RESTful Web Services. REST stands for Representational State Transfer. REST is an architectural style for networked hypermedia applications. It is used to create lightweight web services that can be easily used by any client side language to transfer data.

Views for the web based views had two security layers. i.e. CSRF security layer and Session Based Authentication Layer. CSRF stands for Cross Site Resource Forgery. CSRF security layers use CSRF Tokens which are kind of system generated passwords that are sent with HTTP requests to identify the request and to stop the possible misuse of the system. The idea behind CSRF is that an “attacker (hacker) is ‘forging’ a HTTP request when a victim executes html or JavaScript created by the hacker”. Therefore, those requests are not entertained at the server side. For example, one of the CSRF Tokens generated on our system for a request was TAzB3Ge5Slh82OgynQQvuxGoImJbjGCC. The other security layer that is used for this type of view is the Session Authentication Layer. This layer is used to identify and allow registered users to use the system. Whenever a user logs in, a session is generated for him and the session information is passed with each request

via cookies to the server. This layer is used to prevent any unauthorized user from using the system.

On the REST based views, the security layer used is JWT authentication. JWT stands for JSON Web Token. “JSON Web Token (JWT) is a JSON-based open standard (RFC 7519) for passing claims between parties in web application environment. The tokens are designed to be compact, URL-safe and usable.” On the login endpoint, when the username and password are parsed to the server, the server returns a key that is also like a system generated password. This key is then in the Authorization header and passed with each request. For example, one of the JSON Web Key generated by our system is 9f066e630a0e6d2e0ba1644839ffb3b434589089. On the logout endpoint, the key is deleted, both from the server and the cookies of the device. This layer only allows authorized users to use the application. Django REST framework is used to create the REST based views.

After passing through the security layers, the request finally reaches the appropriate view where logics are implemented. The views process the requests according to the business logic, and after processing the requests, views forward their processed information to the models.

At the models level, classes are declared and the system is based on ORM. ORM stands for Object Relational Model. ORM connects these classes to the database. When a class is declared, a corresponding table is created in the database and database entries are manipulated just like objects are manipulated in the OOP (Object Oriented Programming). The system connects the ORM with the database via configuration settings where it is specified that which Database API is to be used. Database APIs for all the popular databases are placed in the system by default and these are then used to convert the queries written in Models API format to SQL queries. If we need to change the database, we only need to specify the new database API in the settings configuration and the system itself translates all the queries according to the SQL syntax of the new database and thus there is no need to change the queries of whole system just to change the database.

At the database side, we are using PostgreSQL and PostGIS and they are connected through PostGIS API of the system. Pgrouting is not covered by the database API so Raw SQL Querysets are written for them.

2.4 Preliminary Data Preparation

Figure 5 and Figure 6, show flowcharts that have been constructed to illustrate the following explanation:

2.4.1 Data Preparation

Complete and accurate data is a crucial part of intelligent system and have a vital importance in a project like this. It is the data which runs huge systems online. Without data we can confidently say that no system can operate intelligently because with the passage of time it is the data, which, with the aid of different algorithms makes the system more intelligent artificially. Different steps were involved in the data preparation, from acquiring data from open source utilities like OSM towards preprocessing and further processing so that the data should be able to satisfy, at least, minimum requirements, which it did. Data preparation and processing consumed a huge amount of time in our case as any errors in data would let our system do erroneous calculation or sometimes no calculations. The whole process of data preparation consists of different steps which are explained below sequentially.

2.4.2 Data Acquisition

The data used was acquired from OSM (Open Street Map) which is open source or VGI (Volunteered Geographic information) system. In volunteered Geographic information system user upload the data they collect through their mobile or any other device and then upload the data online free for further use. Users who require data can download the data uploaded by other users and use it any purpose, according to their requirement. OSM is a VGI system which is designed for such a purpose on which, users can upload and download from, freely without any cost. Although the data is free, but most of the times the data isn't completely free of errors and uncertainties which must be removed. The data of Islamabad

area was downloaded, which can be downloaded usually by making a polygon and in the area of interest and then exporting all the data in that polygon. Further the data was processed by using different tools and technologies to make it usable.

2.4.3 Importing the data in QGIS

The data coming from Open street maps usually comes in XML (extensible markup language) format and is not directly readable by many softwares and tools. QGIS is an open source GIS utility having support for Open street maps data. By using those tools and technologies, OSM data can be manipulated and different kinds of processing can be performed on that data. QGIS was used to perform some preliminary operations on the data and was then exported to ESRI shapefile format for further processing.

2.4.4 ESRI Shape File

The ESRI Shape file is a well-known GIS vector data format. Developed and maintained by ESRI the data format is used widely for data interoperability between various GIS softwares. It is a GIS data exchange format which is used widely in different tools and technology developed by ESRI. ESRI shape file can also be used in different open source GIS technologies such as QGIS, SAGA GIS, and ENVI etc. ESRI or Environmental System Research Institute is a proprietary software development company which designs a variety of tools and technologies used in geospatial processing. The company is in operation for several decades and has a huge market share in proprietary Geospatial development and analysis tools such ArcMap, ArcView, ArcInfo, ArcObjects etc. The Esri shape file is also coined and developed by ESRI, which is used widely for Geospatial data exchange.

2.4.5 Extracting Roads data

The data acquired from OSM consisted of a lot of information about different entities in the area of interest but we were interested only in roads data which was to be the base of our system. All the data that was not related to roads was removed from the data and only roads data was exported for further processing.

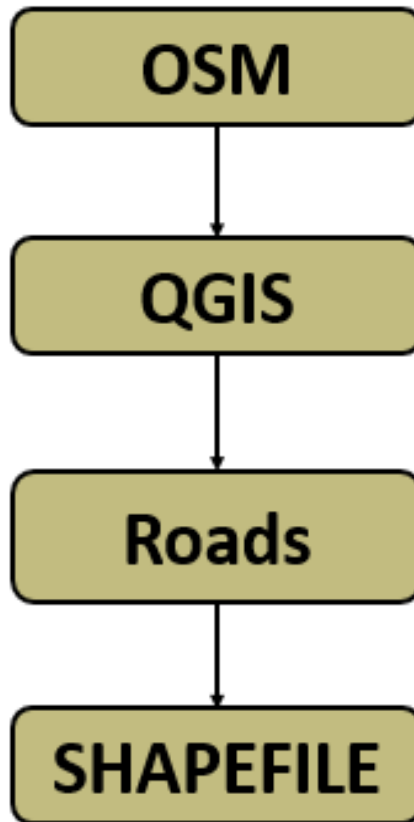


Figure 5: Data Preparation

2.4.6 OGR2OGR

OGR2OGR is a command line utility, a part of GDAL (Geospatial data abstraction library), which is widely used for data conversion from one format to another. Usually different tools and technologies work differently, but few tools and technologies are used for specific purposes, and often accept specific data formats to work with. The command line utility is designed for the purpose to ease the data conversion from one format to another. The beauty of the utility is that it imports the data and can convert data into more than 200 different formats, which is vitally important specifically in the field of Geoinformatics where hundreds of softwares use different data formats, and it is sometime impossible to work on data in some specific format, with different softwares because each software uses a new

data format. Therefore, the command line utilities like OGR2OGR is a blessing for those people who work on different data formats.

2.4.7 PostGIS

PostGIS is a spatial extension of PostgreSQL, a database system used widely for data storage purpose. A database system stores data in different tables connected with each other, the main purpose of which is, efficient data retrieval and proper data storage. Database systems are used widely over the world for storing different kinds of relational and non-relational data. The purpose of using PostGIS is that it comes with a lot of spatial functions which usually don't exist in other databases. In spatial databases, different kinds of spatial operations can be performed on the data, depending on the purpose of the system. The data was then imported to PostGIS and stored in tables, which was prepared for further processing.

2.4.8 Admin Panel (Django)

The admin panel in the model view controller was a user interactive panel which can be used for data manipulation, using user interfaces. By using our admin panel, the data can be manipulated and updated through the use of UI. If there are some errors in the data, they can be corrected through this panel. Missing data can be recreated and updated through the use of that Admin panel. We used the admin panel frequently, for data correction and updating the data. New data was also created by the usage of that panel.

2.4.9 Removing NULL values

As discussed above, the data acquired from OSM was not completely error-free. The obtained data usually contains a lot of errors and uncertainties. Facing similar problem with our data, we had to correct or remove the errors, as they were contaminating the data. In our case, Islamabad's data contained many roads with missing highway type, for example, Highway type can be a highway, Primary road, Secondary road, Tertiary road or Link road, which usually determine the cost. Intuitively, if the highway type of any road is missing, Pgrouting queries cannot work on that data, because Pgrouting calculate cost based on

speed or time required and these parameters are determined by the type of highway. Errors like these were removed both manually and automatically through queries. Errors in some specific data were removed manually one-by-one, while general errors and discrepancies were removed through SQL queries. For example, if the highway type of all or some of the streets are NULL we can remove it by using SQL query like:

update roads set highway='Tertiary'

where name like '%street%'

There were thousands of records in the data and if we start updating each and every street, a huge amount of time would be required to correct the data alone. As we know that streets are usually tertiary highway types, we used queries similar to the above one, to save time, instead of updating each and every record separately. The most difficult, crucial and laborious task was correcting and rectifying the NULL values, as it was difficult to identify highway types, and the surface type, whether it is paved or not. Through digitization admin panel, we were able to view the roads visually and decide subjectively, that, which data must be updated to which type.

2.4.10 Node Network

After removing NULL values and rectifying the data properly, the data was prepared to be used in Pgrouting applications. Creating node network is a function in Pgrouting application which creates different nodes depending on the data. Whenever a road splits into two or more routes, the line segments are broken and new individual line segments are made by node network, to create a way for routing. These line segments are connected with each other through those nodes. Node network of the data was created by using built-in Pgrouting queries. The node network is important to create, as it split the road into nodes and divide them for possible routes in different cases.

2.4.11 Create Topology

The 'create topology' query in Pgrouting converts the node network into vertices. Creating the vertices is a part of Pgrouting for making proper data usable and understandable by Pgrouting queries. The create topology tools were used to make vertices from the node

network data that was made already in the previous step. After create topology tool execution, the data is prepared for running different kinds of queries, consisting of logic from a variety of routing algorithms. Some of the algorithms are tested directly while other are optimized. The algorithm tested in this study were direct and straight forward for two points i.e. source-destination routing. TSP was also tested and gave positive results. The TSP starts from the source point, taking into consideration all the points that come in the way, and then back to the source. Optimized algorithms can also be tested and executed on the data which is not the scope of this study

2.4.12 Pgrouting Queries

In the previous step, by using topology tools, vertices were created for the whole data to make it capable of running Pgrouting queries on. Pgrouting have many algorithms implemented for different routing functions. Shortest path, reduced costs, multiple points and other several algorithms can be used while using Pgrouting. In our case, three algorithms were tested which were:

2.4.12.1 Travelling sales man problem for multiple points

The TSP or travelling salesman problem is an algorithm in computer science use for finding optimal route containing a large number of points. In travelling sales man problem, the route starts from a source and go through all the points in the route and come back towards the destination after completing the cycle. The TSP is widely used in routing application especially in delivery services where a deliverer has to deliver some products on multiple points and then come back towards destination. We have to optimize the algorithm along with the manipulation of the results. As we don't need the driver to come back towards the origin, the results need to manipulate for not coming to origin again.

2.4.12.2 Dijkstra for shortest path between two points

Dijkstra algorithm is an algorithm in computer sciences which is also implemented in Pgrouting, is used to find out the optimal route between two points which is a short distance reduced cost. The cost can be in terms of distance, speed and travel time. For example, if highway type of a road is primary, it is believed through common sense that the speed of

vehicle will be higher as compared to the speed at some other secondary or tertiary roads. Speed values were given to the roads on the basis of subjective findings. The algorithm starts from the source node and solve each node directly connected to the source node in the first step. After first step, all unsolved nodes connected to the solved nodes are solved and marked as solved. As the solved nodes are marked as solved all the subsequent nodes connected to that node are solved. After solving all the nodes, different path connected from source to destination are marked and summed. The route having the lowest cost is selected further. Dijkstra works by analyzing vertices in such a manner in which the first vertex is examined and solved for all closest vertices and thus going in towards the end vertex. The algorithm work in a manner looking all the points from start to end. For small data the Dijkstra works better but for large number of points in the way the algorithm takes a lot of time going towards a worst performance.

2.4.12.3 A* for two points

A* algorithms is same like Dijkstra in manner that both algorithms are used to find the shortest optimal route between two points. A* algorithm is an alternative for Dijkstra when Dijkstra doesn't work well. A* is in the class of greedy algorithm which works in heuristic manner as compared to the all points approach of Dijkstra. A* have very good performance for larger data but Dijkstra work well than A* when the amount of data is less or number of vertices in the path reduced. A* is usually called a complete algorithm which means it will found out the result, if they exist. The process of searching is speeded up by using heuristic, which is the main advantage of A* over Dijkstra. One of the main advantage of A* algorithm is, it is able to search in multiple directions.

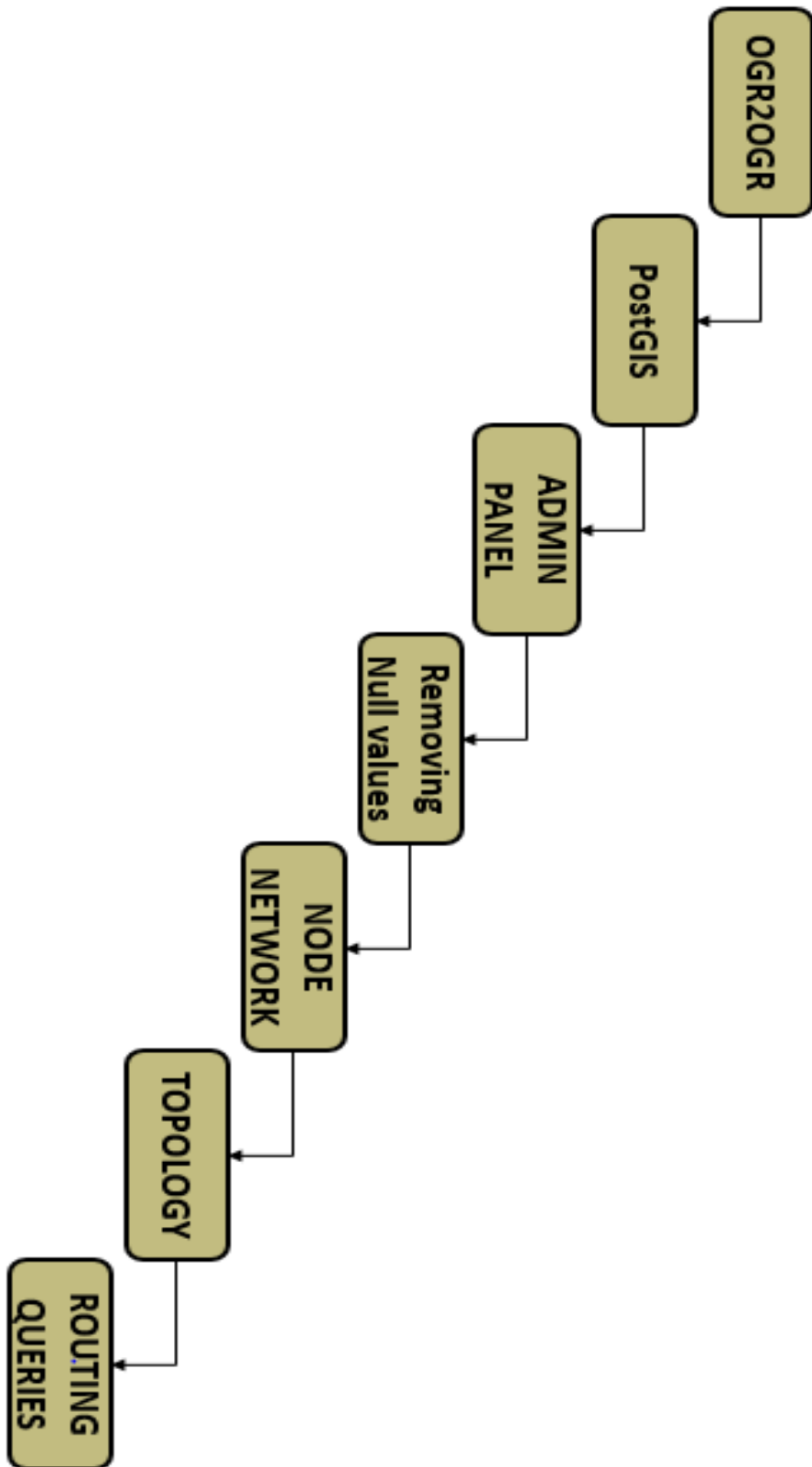


Figure 6: Creating Routing System

2.5 Algorithm Development

2.5.1 Self-Learning

The following flow diagram in Figure 7, represents the machine learning part of the system:

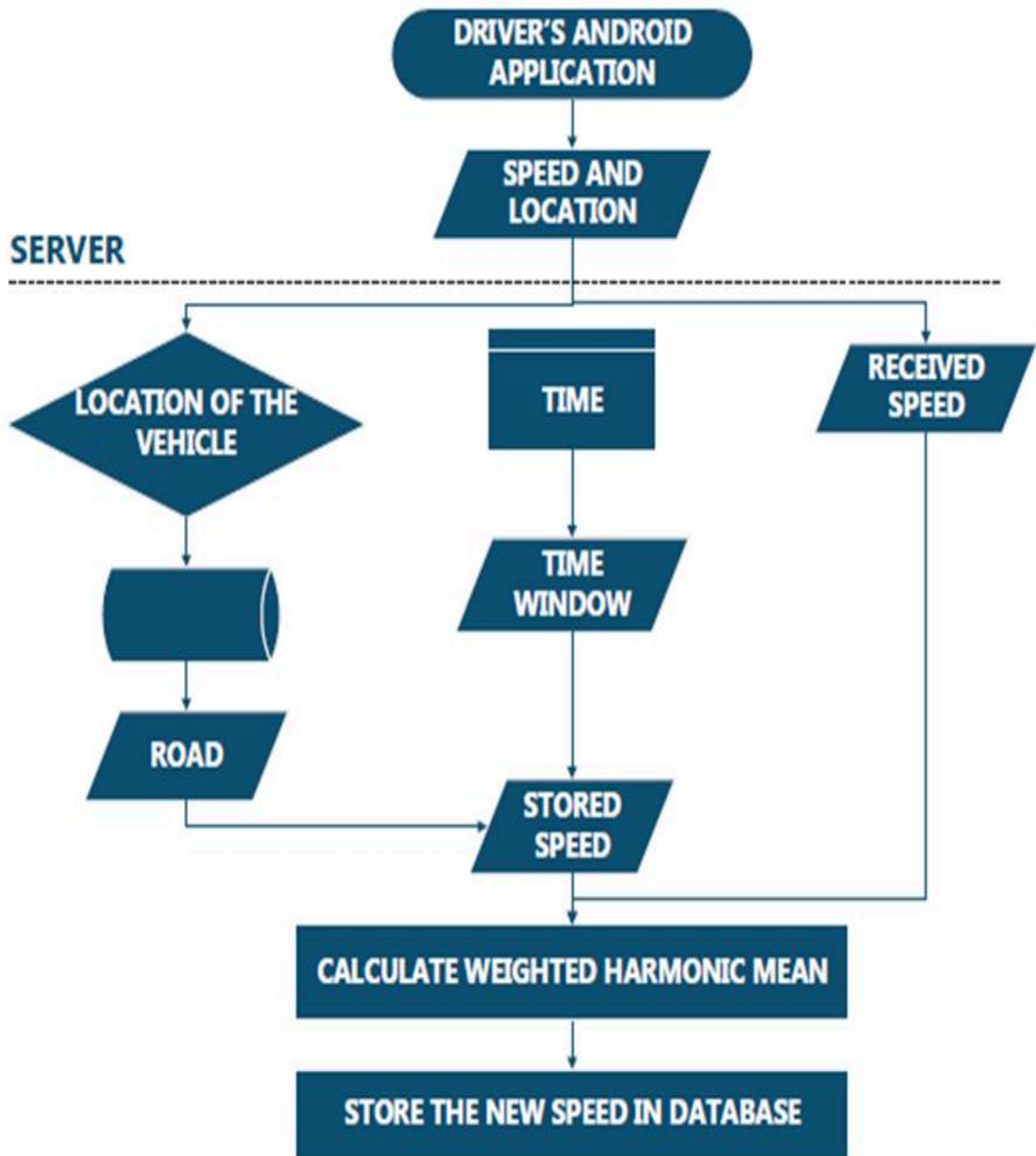


Figure 7: Self Learning of road networks

This flow was incorporated to first acquire and then enhance the quality of the Spatio-Temporal data of road networks present in the system. Traffic pattern is a spatio-temporal phenomenon. i.e. it changes with time. Density of traffic on a road might be very high at daytime and the same road might be completely empty at night. Each day is divided into 96 time windows, i.e. each hour is divided into 4 quarters. These time windows save the information and speed information is classified into these time windows.

Figure 8 explains that the driver's android application calculates the current speed of the vehicle and transmits it along with the current location of the driver to the server after every 5 seconds, if the driver is on duty.



Figure 8: Location and speed calculated every 5 seconds

Figure 9 explains that the speed and location information that is collected every five seconds on the driver's application is then sent to the server, the moment it is collected.

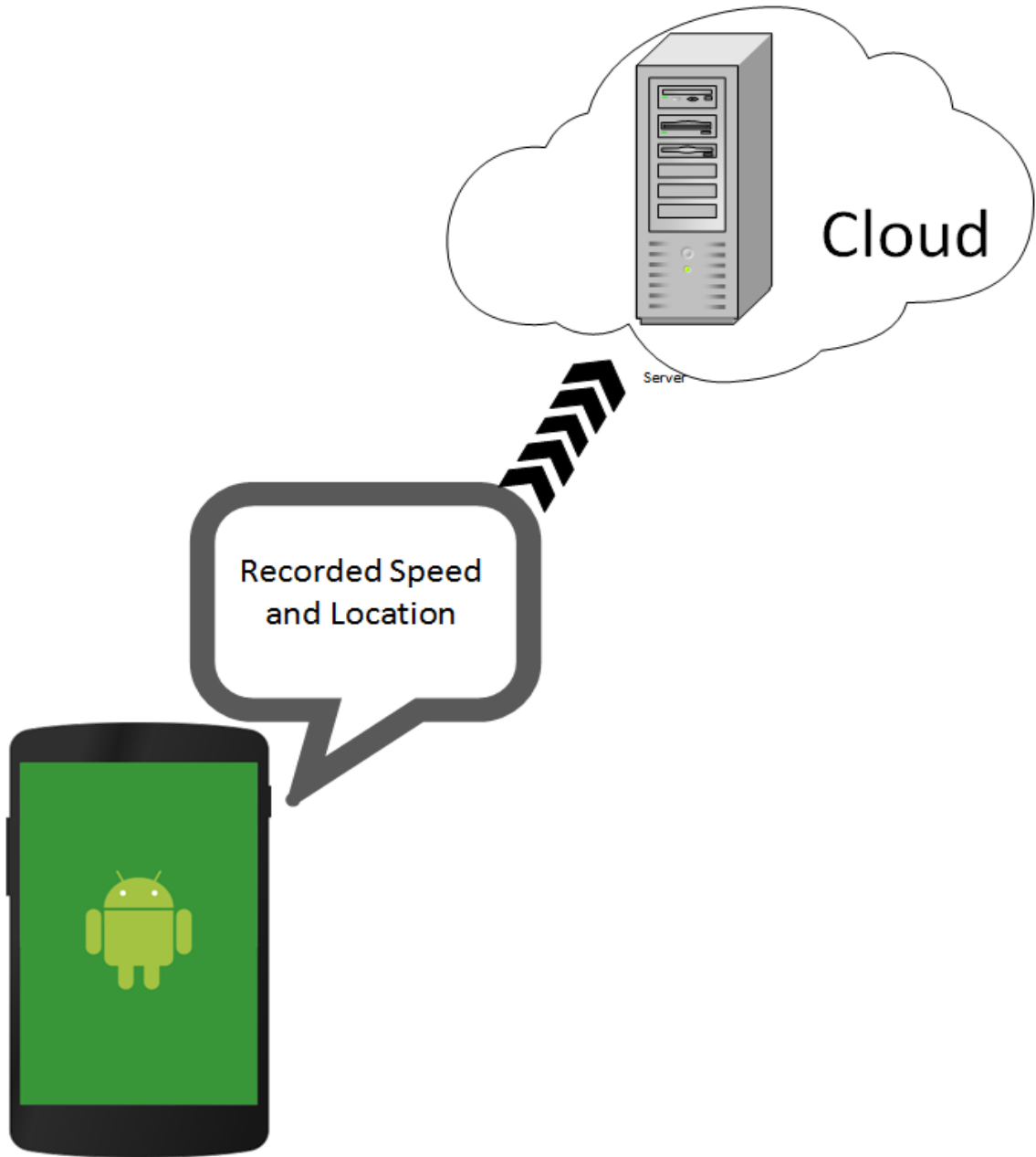


Figure 9: Location and speed sent to server

Figure 10 explains that when the request arrives at the server, a time window is assigned to the information according to the current time.

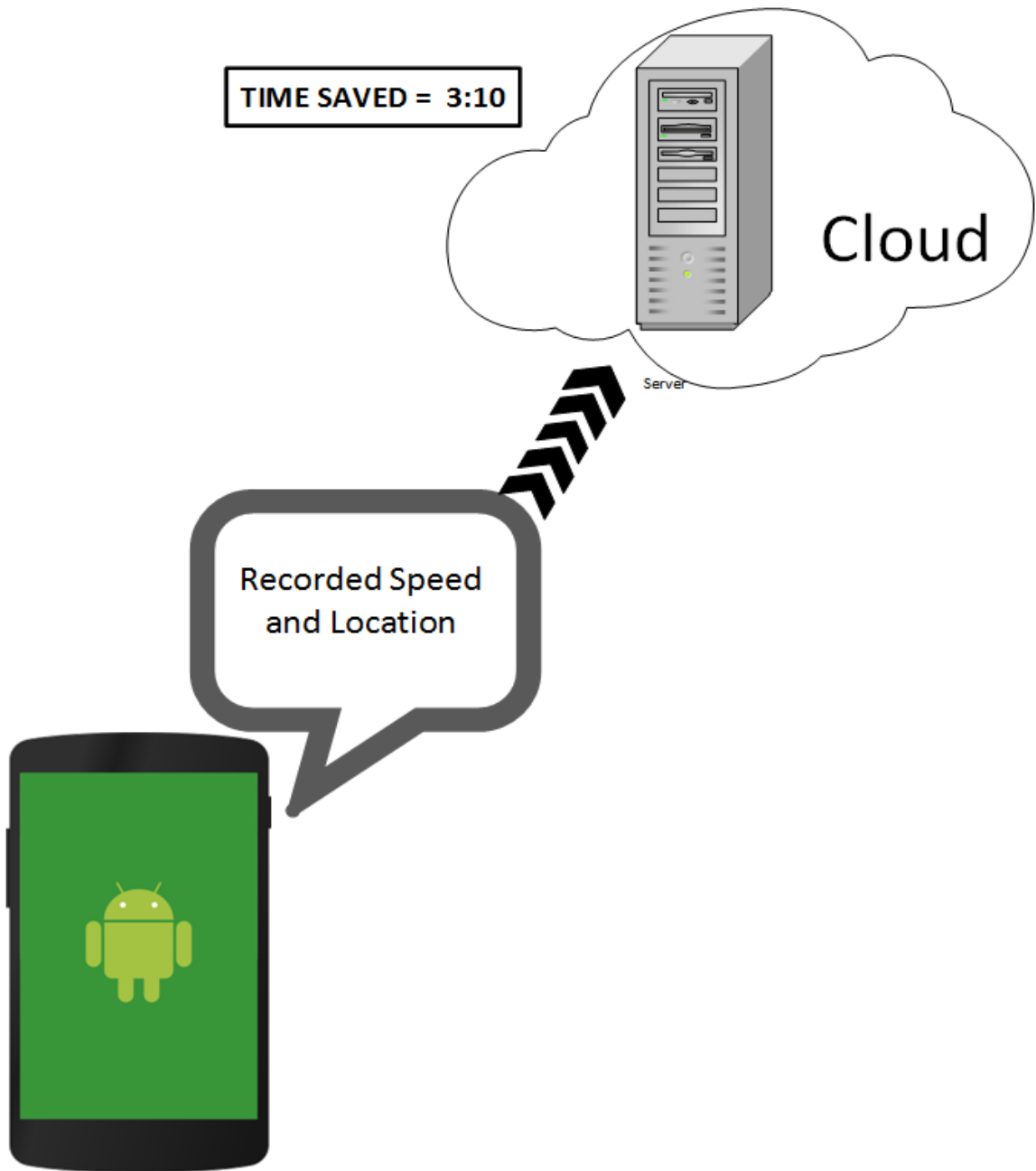
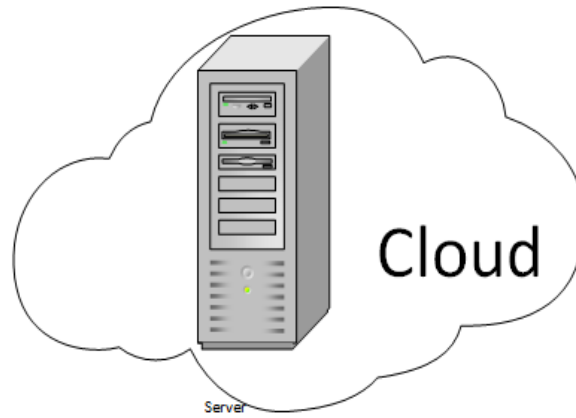


Figure 10: Timestamp assigned to information received at server

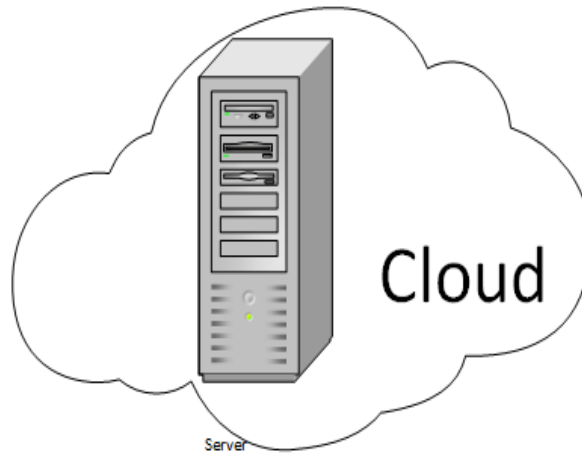
Figure 11 explains that the location information is used to select the particular road on which the vehicle is moving currently.



No.	Time window	Road	Speed
1.	1 (12:00 a.m-12:15 a.m)	Kashmir Highway	7 m/s
2.	2 (12:15 a.m-12:30 a.m)	Kashmir Highway	7 m/s
61.	61 (3:00 p.m-3:15 p.m)	Kashmir Highway	5 m/s

Figure 11: Road selected from Location Information

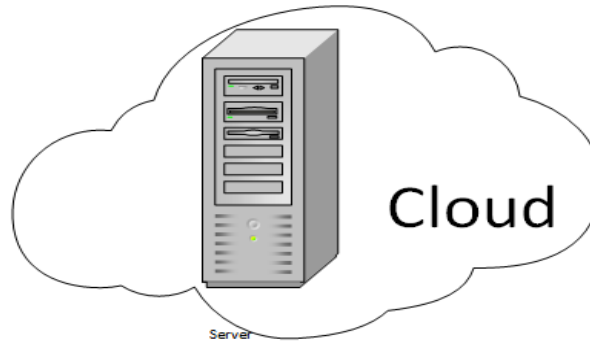
Figure 12 explains that the value of speed of the current time window of the road on which the vehicle is moving is then updated using weighted harmonic mean.



No.	Time Saved	Time window	Location	Road	Recorded Speed	Speed in Time Window	Weighted harmonic mean
1.	3:10 p.m.	61 (3:00-3:15)	33.6762437, 73.0298655	Kashmir Highway	4 m/s	5 m/s	4.44
2.							
3.							
4.							

Figure 12: Harmonic mean taken

Figure 13 explains that weighted harmonic mean is used because it removes the outliers while aggregating the speed information. The current speed that is sent to the server is given a weight of one while the speed stored in the system is given a weight equal to the count of previous updates on that road on the current time window. These weights are then used to calculate harmonic mean. Speed and new count is then updated in the system.



UPDATED TABLE

No.	Time window	Road	Speed
1.	1 (12:00 a.m-12:15 a.m)	Kashmir Highway	7 m/s
2.	2 (12:15 a.m-12:30 a.m)	Kashmir Highway	7 m/s
61.	61 (3:00 p.m-3:15 p.m)	Kashmir Highway	4.44 m/s

Figure 13: Values updated using harmonic mean

This information can then be used to give a more accurate and more dynamic estimation of the time needed to complete a trip. Time window, for a better understanding, is a time period, which is in this case divided into 15 minutes. Each division of 15 minutes of the day is allotted a speed, according to the taxi's speed and it then dynamically maps the traffic density of each time window.

2.5.2 Ride Matching Algorithm

The following flow diagram in Figure 14, gives a detailed illustration of the algorithm used to match rides:

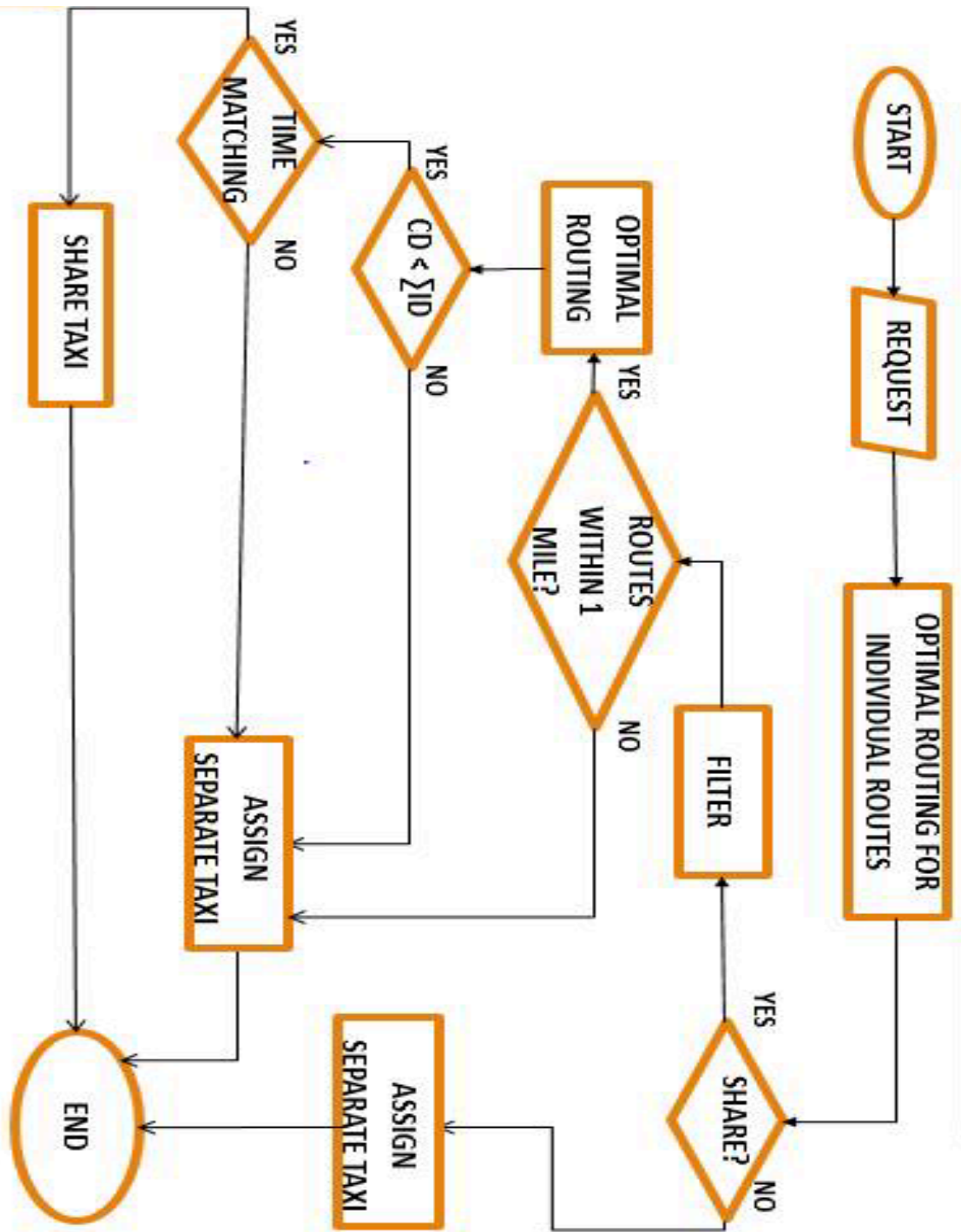


Figure 14: Ride Matching Algorithm

The algorithm is dynamic enough to process the requests that are received on the server within next 10 minutes. The algorithm shown in Figure14 runs every 10 minutes.

When a request arrives at the server, it already has origin and destination points with it. Origin and destination are then matched and the best possible route for the two points is made and saved along with the request in the database. The request also sends the information about the user that whether the user is ready to share the ride or not.

The request also contains the earliest departure time and the latest arrival time for the request. This information is also sent by the user. This gives the algorithm an idea about the time window that has to be considered while planning the shared trip.

After every 10 minutes, when the ride matching algorithm is called, it checks for the ride requests within next 15 minutes. For each request, it makes the same kind of checks:

First, it sees if the user is asking for a ride share or not. In case of 'no', the algorithm assigns a separate taxi to that request. If the user wants to share, it moves to the next step.

In the next step, the system checks if there are any other requests in the next 15 minutes which have their routes within 1 mile of this request. If there is no other request in the next 15 minutes with its route in the proximity of 1 mile of the route of this request, it means that there is no other route that is closer to this one so a separate taxi for this route is assigned to this request. If there is another request in the next 15 minutes with its route in the proximity of 1 mile of the route of this request, it means that the two requests are similar and the two requests are sent to the next step.

In the next step, we see if the total number of seats required is more than three. If it is more than three, ride is not shared because usually there are only 3 seats in a taxi.

In the next step, we check which origin has to be reached earlier and which destination has to be reached latest. This is checked from the time window information that comes with the request. The earliest origin and latest departure points are taken as origin and destination respectively and the other two points are taken as way points and a shared route is calculated using the TSP algorithm. The duration of this route is then calculated.

Then the duration of the shared path is compared to the sum of durations of the individual paths of the two requests. If the duration of shared path is more than the sum of duration of individual paths, it means that the ride sharing is not feasible and total system wide time

is increasing. If the duration of shared route is less than the duration of sum of individual routes, it means that total system wide time is decreasing and sharing the ride will be feasible.

Therefore, in case the duration of shared route is less than the sum of duration of individual routes, ride sharing is feasible. Otherwise, separate taxis are assigned to both the requests because ride sharing is not feasible in this case.

In the next step, we check if the total duration of the shared routes satisfies the time windows of both the requests. i.e. no user has to arrive after the latest arrival time. If the time windows are not satisfied, rides are not shared, and separate taxis are assigned to both the requests for their individual routes. If the time windows are satisfied, rides are shared and a single taxi is assigned to both requests for their shared paths.

The above steps are repeated for each request in the next 15 minutes and the whole algorithm is recalled after every 10 minutes hence making it possible to process each ride request within 10 minutes.

2.6 Data Model Development

ER Diagram

An Entity-Relationship model or ER Diagram, helps define the interrelated things of interest in a specific area of data. It is composed of entity types and specifies relationships that may exist between instances of those entity types. Figure 15 shows the ER diagram of all the entities required for the formation of tables of our data.

It explains which the entities are linked together, which entities are dependent on the other, which are independent, and how they are linked. It can be helpful to understand the working of the project, and about the data required to carry the operations and functions on.

In this project, the USER entity is the main element, that directly or indirectly connects to most of the other entities. For further understanding, take a look at this diagram:

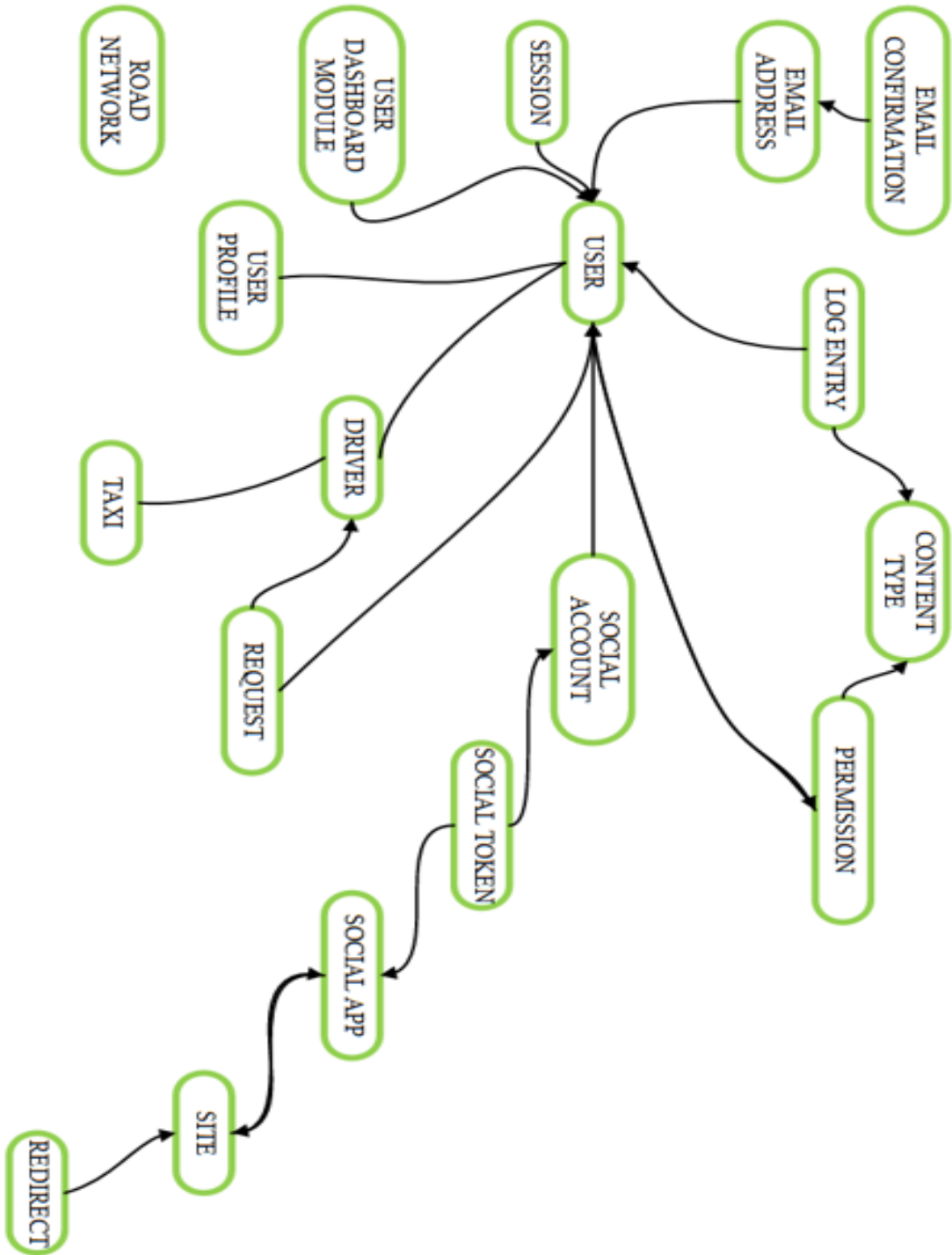


Figure 15: ER Diagram

RESULTS

A system like this can make daily commute much better and easier. It will save user's money, time and fuel and making the environment greener. Machine learning capabilities make the system intelligent enough to help it get better with time.

Following are very few the snapshots of our system's prototype:

Chalti Gari Home About Contact Account ▾

Sign In

Please sign in with one of your existing third party accounts. Or, [sign up](#) for a Chalti Gari account and sign in below:

- Twitter
- Google
- Facebook

or

Username: ramiz

Password: *****

Remember Me:

[Forgot Password?](#)

Figure 16: Sign in page

Chalti Gari Home About Contact Account ▾

Sign Up

Already have an account? Then please [sign in](#).

Username: Username

E-mail: E-mail address


Password: Password

Password (again): Password (again)

Figure 17: Signup Page

Edit your profile here

Picture*

Currently: 

Change:

No file chosen

Gender

Male

Phone number

+923318873370

Address

house # 162-A, Street # 36-A, G-9/1, Islamabad

First name

Syed Ramiz

Last name

Sami

Figure 18: Edit profile page

HOME > SITE ADMINISTRATION

widgets + Syed Ramiz

You have signed out.

Successfully signed in as ramiz.

SECTIONS	RECENT ACTIONS	ADMINISTRATION
ACCOUNTS Email addresses Email confirmations AUTH TOKEN Tokens DIGITIZERROADS Road networks REDIRECTS Redirects REGISTER Sign ups SITES Sites SOCIAL ACCOUNTS Social accounts	X Request Chaltigari Taxi abc123 Taxi abc123 Road network Pitras Bukhari Road Road network Service Road West G7 Road network Unknown Road Road network Service Road East G8 Road network Unknown Road Road network Unknown Road Road network Unknown Road	AUTHENTICATION AND AUTHORIZATION Groups Users QUICK LINKS Return to site Change password Log out

powered by DJANGO JET

Figure 19: Admin home page

HOME
VIEW SITE
DOCUMENTATION

APPLICATIONS
Accounts
Auth Token
Authentication and Authoriza...
Digitizeroads
Redirects
Register
Sites
Social Accounts
Taxis
Userprofile

BOOKMARKS

powered by DJANGO JET

Search name highway

oneway surface inserted updated

GID	NAME	HIGHWAY	ONEWAY	SURFACE	INSERTED	UPDATED
2227	Jinnah Avenue	secondary	yes	paved	Empty	May 16, 2016, 3:28 p.m.
2226	Jinnah Avenue	secondary	yes	paved	Empty	May 16, 2016, 3:28 p.m.
2225	Faisal Avenue	secondary	yes	paved	Empty	May 23, 2016, 4:03 p.m.
2224	Faisal Avenue Underpass	secondary	yes	paved	Empty	May 23, 2016, 4:05 p.m.
2223	Faisal Avenue Underpass	secondary	yes	paved	Empty	May 23, 2016, 4:05 p.m.
2222	Faisal Avenue Underpass	secondary	yes	paved	Empty	May 23, 2016, 4:05 p.m.
2221	Faisal Avenue Underpass	secondary	yes	paved	Empty	May 23, 2016, 4:05 p.m.
2220	Link Road	secondary	no	paved	Empty	June 3, 2016, 5:38 p.m.
2219	Link Road	secondary	no	paved	Empty	June 3, 2016, 5:40 p.m.

GO 0 of 100 selected 1951 road networks 1 2 3 4 ... 19 20

Figure 20: Roads editing page

HOME
VIEW SITE
DOCUMENTATION

APPLICATIONS
Accounts
Auth Token
Authentication and Authoriza...
Digitizeroads
Redirects
Register
Sites
Social Accounts
Taxis
Userprofile

BOOKMARKS

powered by DJANGO JET

Gid:*

Geom:*

Name:*

Highway:*

Oneway:*

Surface:*

Figure 21: Page to digitize roads

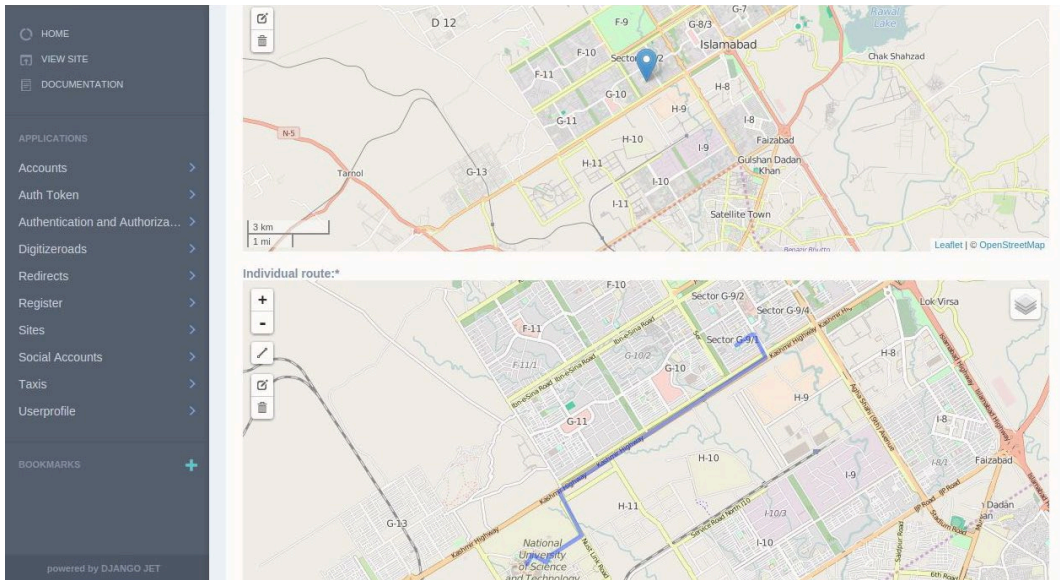


Figure 22: A request, as viewed in the admin panel

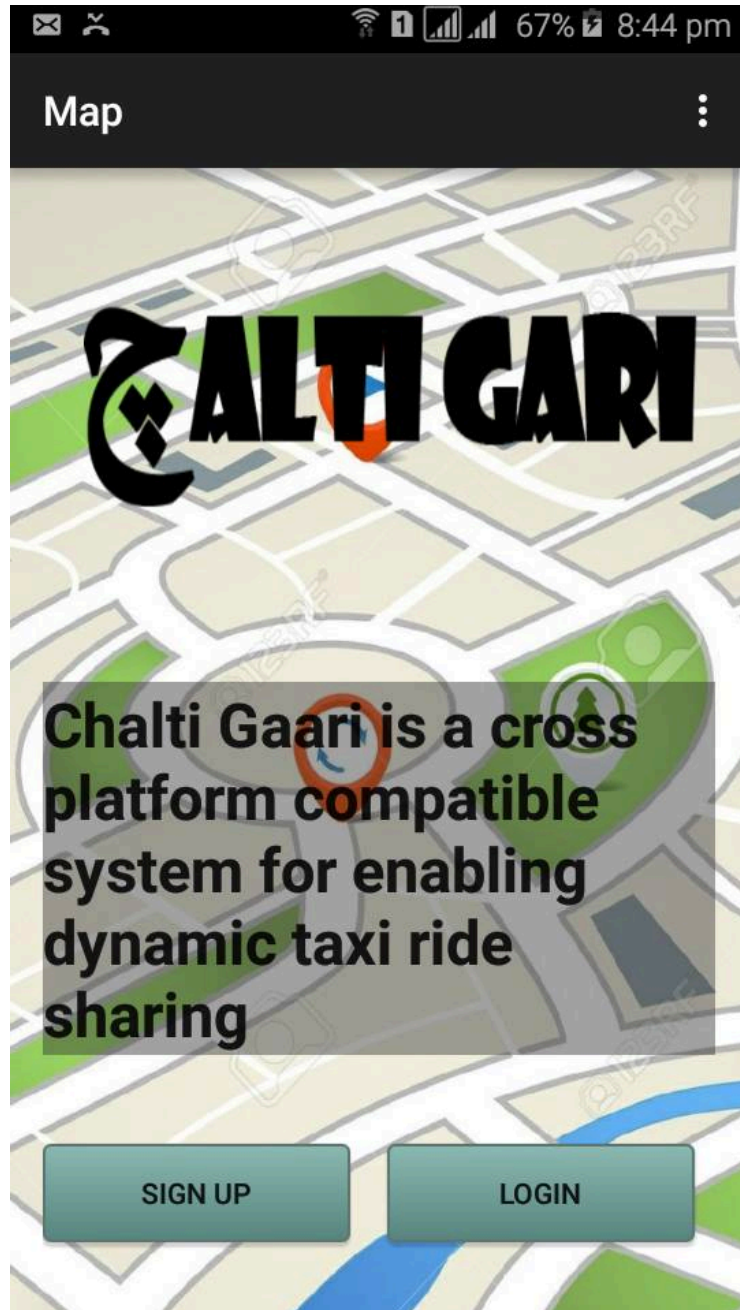


Figure 23: Application Homepage

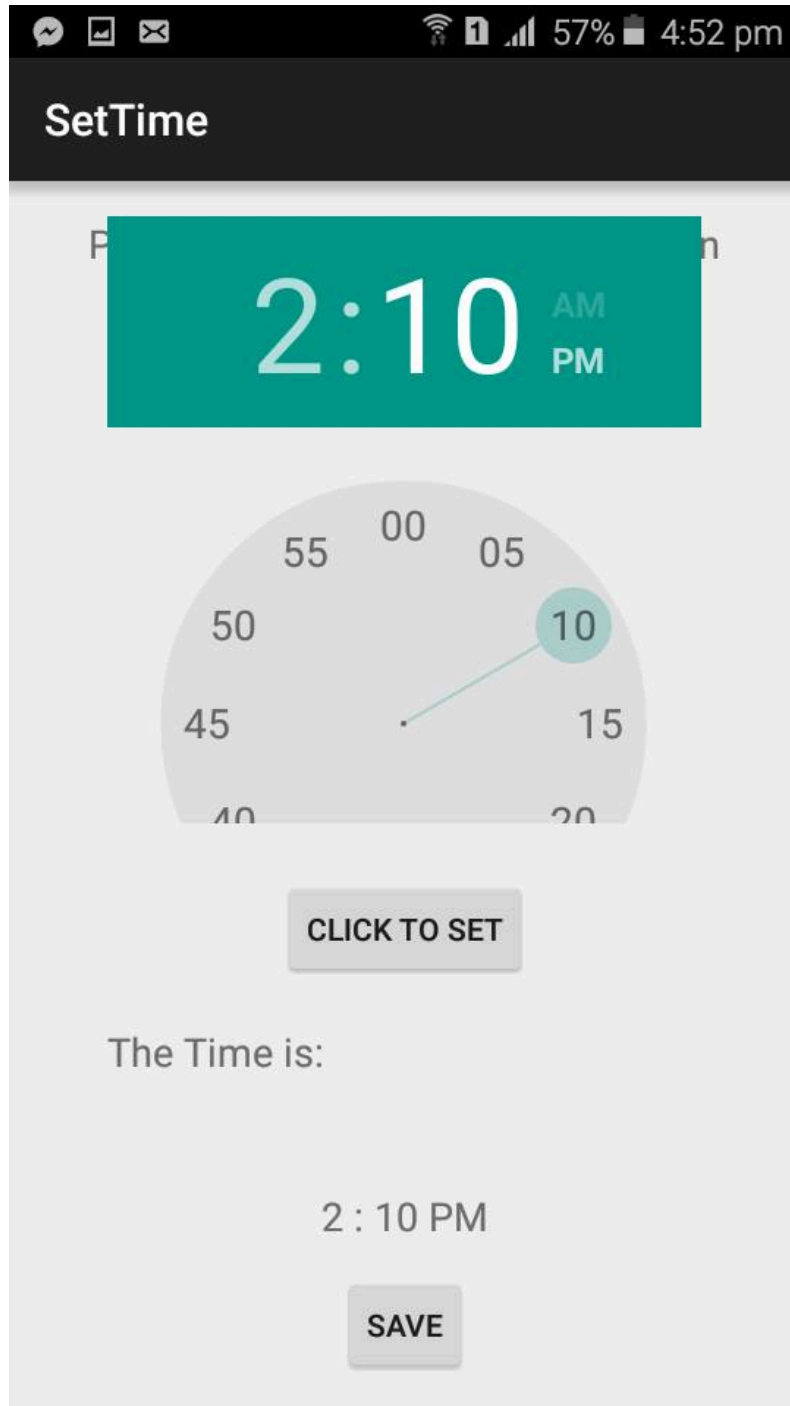


Figure 24: Setting time on the application for time window

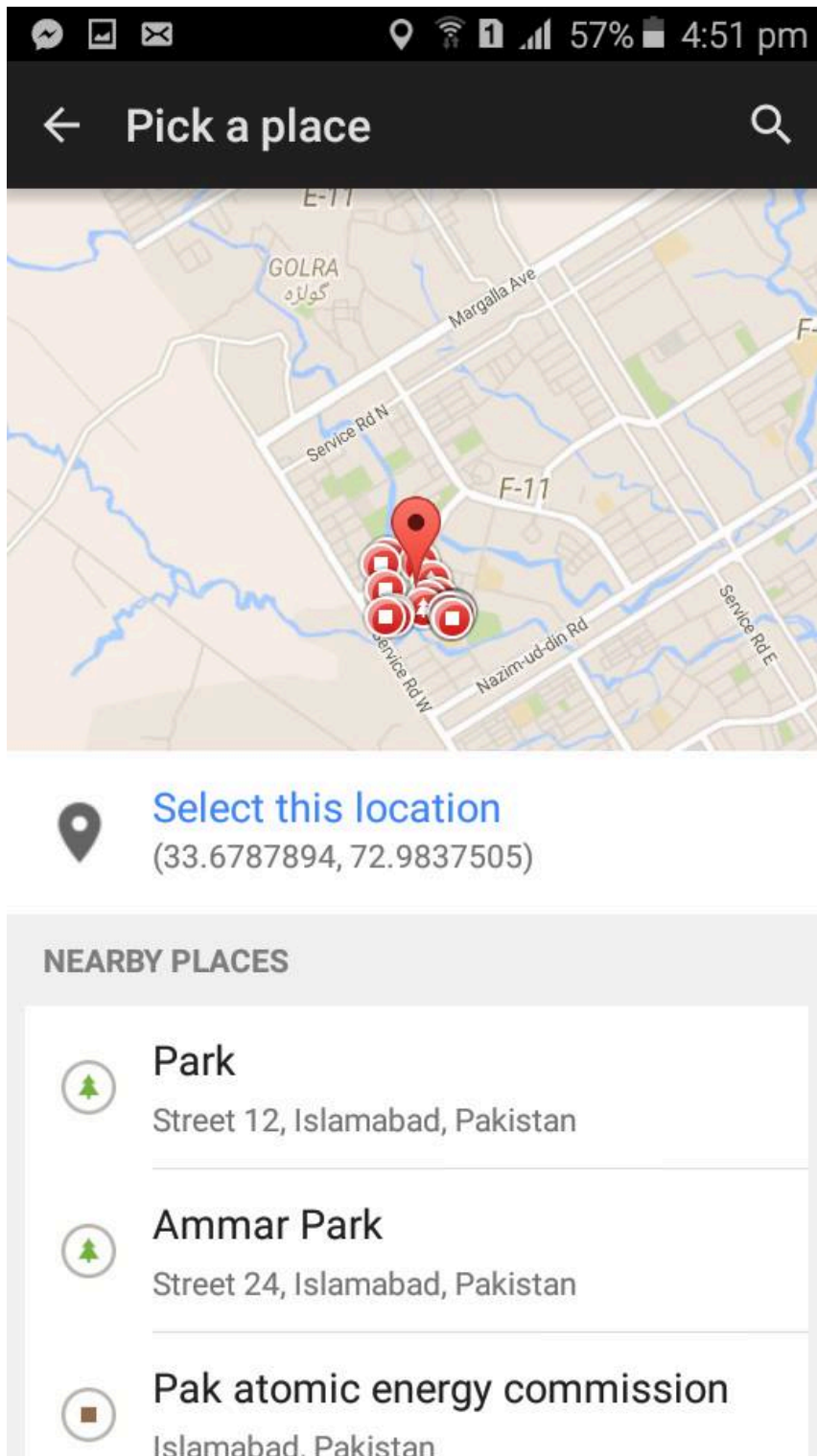


Figure 25: Picking origin and destination points on the application

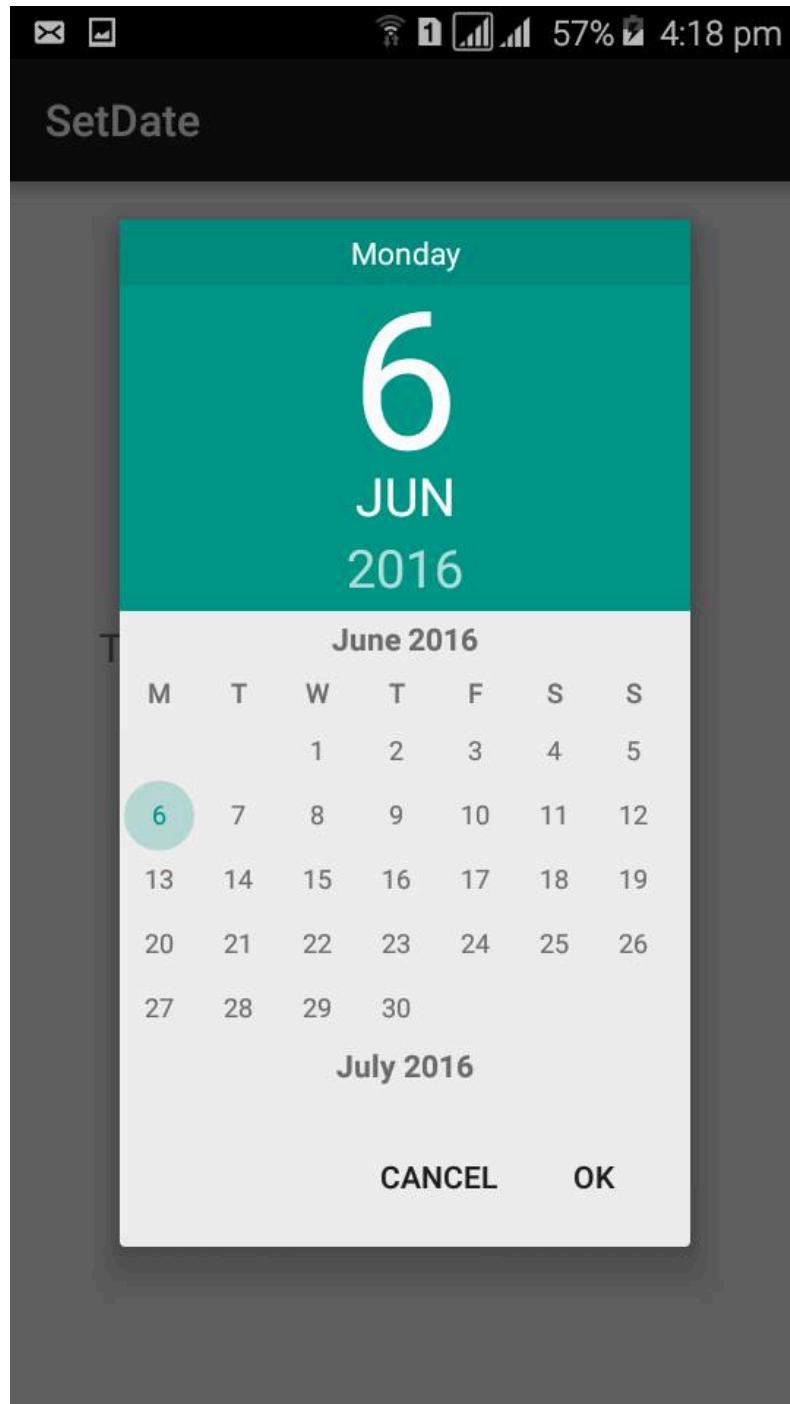


Figure 26: Scheduling rides on the application

RECOMMENDATIONS

The following are the recommendations as well as our future plans to convert this prototype into a product and take the project to the market and take it to the next level:

- 1) Getting on field speed data for enhancement of road network
- 2) Work on the UX according to user feedback
- 3) Optimizing pgrouting queries for performance
- 4) Starting the system on some professional routing service like Google's directions API and then Shifting to pgrouting using native data after enough data has been gathered from the field and the data is good enough.

REFERENCES

- 1) Agatz, N., Erera, A., Savelsbergh, M., & Wang, X. (2010). Sustainable passenger transportation: Dynamic ride-sharing.
- 2) Agatz, N., Erera, A., Savelsbergh, M., & Wang, X. (2012). Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research*, 223(2), 295-303.
- 3) Agatz, N. A., Erera, A. L., Savelsbergh, M. W., & Wang, X. (2011). Dynamic ride-sharing: A simulation study in metro Atlanta. *Transportation Research Part B: Methodological*, 45(9), 1450-1464.
- 4) Ma, S., Zheng, Y., & Wolfson, O. (2013, April). T-share: A large-scale dynamic taxi ridesharing service. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on* (pp. 410-421). IEEE.
- 5) Tao, C.C. (2007, September). Dynamic taxi-sharing service using intelligent transportation system technologies. In *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on* (pp. 3209-3212). IEEE.
- 6) Zhang, L., & He, X. (2012). Route Search Base on pgRouting. In *Software Engineering and Knowledge Engineering: Theory and Practice* (pp. 1003-1007). Springer Berlin Heidelberg.
- 7) Django Documentation: <https://docs.djangoproject.com/en/1.9>
- 8) Django vs Laravel vs Rails: <http://www.findalltogether.com/post/django-vs-laravel-vs-rails/>
- 9) CMS vs MVC: <http://www.findalltogether.com/post/cms-vs-mvc-frameworks/>
- 10) Python vs PHP vs Ruby: <http://www.findalltogether.com/post/python-vs-php-vs-ruby/>